This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A structure recovery system comprising: analysis means for analyzing the structure of a data string written in accordance with a predetermined rule , said rule comprising:

a) generating a grammar information object;

b) emptying buffers to receive a data string;

c) causing a context pointer to indicate a root node of an abstract syntax tree;

d) obtaining a set of syntax recovery units;

e) inputting a data string comprising a stream of tokens to the buffers;

f) extracting a token from the buffer as a target token;

and when the target token is the end of the data string generating and outputting an abstract data tree, and for detecting an error in accordance with said predetermined rule by a method further comprising receiving a non-terminal token, determining from the grammar information object if the token matches a context pointer, and if so adding the token to the context pointer by the steps of:

a) adding a node to which the terminal token is appended to the context pointer;

b) shifting the point indicated by the by the context pointer to the newly added node to which the non-terminal token is appended;

c) repeating steps a) and b) for all child nodes;

d) shifting the context pointer to the parent node; and

e) obtaining a next token if the node does not contain all child nodes or if the context pointer points back to a parent node; and recovery means for, upon the receipt of a request from said analysis means, correcting in accordance with said predetermined rule said error detected in said data string by said analysis means, wherein said recovery means includes a set of ~~correction means~~ syntax recovery units that individually employ simple functions for correcting specific types of errors, and wherein said recovery means selectively employs said syntax recovery ~~correction~~ means based on the error type in

accordance with said predetermined rule in order to correct a variety of errors in said data string by a method further comprising sequentially using syntax recovery units where a syntax recovery unit receives an error message if a token does not grammatically match the syntax pointer, and the syntax recovery unit sequentially passing the error to a subsequent syntax recovery unit until a syntax recovery unit matching the error type is found, said syntax recovery unit

a) resetting the current abstract syntax tree, buffer, and context pointer;

b) recovering a pointer where in accordance with said syntax rules no error exists; and

c) notifying the parser that the correction was successfully applied.

2. (Canceled)

3. (Currently Amended) The parsing system according to claim [[2]] 1, wherein multiple types of said syntax recovery units are prepared in accordance with the types of errors that are detected by said parser in said data string, and each of said syntax recovery units has a function for correcting a specific type of error.

4. (Original) The parsing system according to claim 3, further comprising: corresponding information storage means for storing information that correlates the type of data string with a syntax recovery unit for recovering from an error in said data string, wherein, in accordance with the type of target data string, said parser employs said information stored in said corresponding information storage means to set up said syntax recovery unit for the correction of an error upon the receipt of a request.

5. (Original) The parsing system according to claim 3, wherein, when said target data string includes an element that is not defined by a rule that said parser employs for said parsing process, at least one of said syntax recovery units is activated and performs a process for replacing said rule used by said parser with a rule that defines said element in said target data string, and for returning said target data string to said parser.

6. (Original) The parsing system according to claim 2, further comprising: a lexical analyzer, for performing token analysis for said target data string; and a token recovery unit, for correcting an error detected by said lexical analyzer in said token in said data string, wherein said token recovery unit can change the contents of a correction.

7. (Original) The parsing system according to claim 6, wherein multiple types of said token recovery units are prepared in accordance with the type of error that is detected by said lexical analyzer in said data string, and each has a function for correcting a specific type of error.

8. (Currently Amended) A system for converting a data string in a predetermined form into a data string in another form comprising: an analyzer for analyzing said data string; said analyzer comprising

a) means generating a grammar information object;

b) means for emptying buffers to receive a data string into the system;

c) means for causing a context pointer to indicate a root node of an abstract syntax tree;

d) means for obtaining a set of syntax recovery units;

e) means for inputting a data string comprising a stream of tokens to the buffers of the system;

f) means for extracting a token from the buffer as a target token;

and means when the target token is the end of the data string for generating and outputting an abstract data tree,

means for detecting an error in accordance with said predetermined rule by a method further comprising receiving a non-terminal token, determining from the grammar information object if the token matches a context pointer, and if so adding the token to the context pointer by the steps of:

a) adding a node to which the terminal token is appended to the context pointer;

b) shifting the point indicated by the by the context pointer to the newly added node to which the non-terminal token is appended;

c) repeating steps a) and b) for all child nodes;

d) shifting the context pointer to the parent node; and

e) obtaining a next token if the node does not contain all child nodes or if the context pointer points back to a parent node;

a recovery ~~unit~~ means comprising a set of syntax recovery units that individually employ simple functions for correcting specific types of errors, and wherein said recovery means selectively employs said syntax recovery means based on the error type in accordance with said predetermined rule in order to correct a variety of errors in said data string by a method further comprising sequentially using syntax recovery units where a syntax recovery unit receives an error message if a token does not grammatically match the syntax pointer, and the syntax recovery unit sequentially passing the error to a subsequent syntax recovery unit until a syntax recovery unit matching the error type is found, said syntax recovery unit

a) resetting the current abstract syntax tree, buffer, and context pointer;

b) recovering a pointer where in accordance with said syntax rules no error exists; and

c) notifying the parser that the correction was successfully applied ~~for~~, said syntax recovery unit upon the receipt of a request from said analyzer, correcting an error detected in said data string by said analyzer; and a converter, for changing a data form in accordance with the results obtained by said analyzer, wherein multiple types of said recovery units are prepared in accordance with the type of error that is detected by said analyzer in said data string, and each has a function for correcting a specific type of error.

9. (Currently Amended)) The conversion system according to claim 8, wherein said analyzer is a parsing means for parsing said data string, and said recovery unit is syntax recovery means for correcting an error in said data string in accordance with a syntax rule.

10. (Currently Amended) A computer comprising: an input unit for receiving a data string written in accordance with a predetermined rule, said rule comprising:

a) generating a grammar information object;

b) emptying buffers to receive a data string;

c) causing a context pointer to indicate a root node of an abstract syntax tree;

d) obtaining a set of syntax recovery units;

e) inputting a data string comprising a stream of tokens to the buffers;

f) extracting a token from the buffer as a target token;

and when the target token is the end of the data string generating and outputting an abstract data tree; a processor for processing said data string by using a function implemented by program control; and an output unit for outputting said data string obtained by said processor, wherein said processor includes an analyzer for analyzing said data string including receiving a non-terminal token, determining from the grammar information object if the token matches a context pointer, and if so adding the token to the context pointer by the steps of:

a) adding a node to which the terminal token is appended to the context pointer;

b) shifting the point indicated by the by the context pointer to the newly added node to which the non-terminal token is appended;

c) repeating steps a) and b) for all child nodes;

d) shifting the context pointer to the parent node; and

e) obtaining a next token if the node does not contain all child nodes or if the context pointer points back to a parent node; and a syntax recovery unit, for, upon the receipt of a request from said analyzer, correcting an error detected in said data string by said analyzer, and wherein multiple types of said syntax recovery units are prepared in accordance with the type of error that is detected by said analyzer in said data string, and each has a function for correcting a specific type of error wherein said recovery means includes a set of syntax recovery units that individually employ simple functions for correcting specific types of errors, and wherein said recovery means selectively employs said syntax recovery units based on the error type in accordance with said predetermined rule in order to correct a variety of errors in said data string by a method further comprising sequentially using syntax recovery units where a syntax recovery unit receives an error message if a token does not grammatically match the syntax pointer, and the syntax recovery unit sequentially passing the error to a subsequent syntax recovery unit until a syntax recovery unit matching the error type is found, said syntax recovery unit

a) resetting the current abstract syntax tree, buffer, and context pointer;

b) recovering a pointer where in accordance with said syntax rules no error exists; and

c) notifying the parser that the correction was successfully applied.

11. (Currently Amended) The computer according to claim 10, wherein said analyzer is a parsing means for parsing said data string, and said recovery unit is syntax recovery means for correcting an error in said data string in accordance with a syntax rule.

12. (Currently Amended) A parsing method for parsing a data string written in accordance with a predetermined rule comprising the steps of: selecting a program module used to correct an error in a target data string in accordance with a syntax rule; parsing said data string by inputting a data string comprising a stream of tokens to associated buffers and extracting a token from the buffer as a target token, and when the target token is the end of the data string generating and outputting an abstract data tree, and detecting an error in accordance with said predetermined rule by receiving a non-terminal token, determining from the grammar information object if the token matches a context pointer, and if so adding the token to the context pointer, and otherwise issuing a correction request to said program module when said parsing detects an error in accordance with said syntax rule in said data string by the steps of:

a) adding a node to which the terminal token is appended to the context pointer;

b) shifting the point indicated by the by the context pointer to the newly added node to which the non-terminal token is appended;

c) repeating steps a) and b) for all child nodes;

d) shifting the context pointer to the parent node; and

e) obtaining a next token if the node does not contain all child nodes or if the context pointer points back to a parent node; and recovery means for, upon the receipt of a request from said analysis means, correcting in accordance with said predetermined rule said error detected in said data string by said analysis means, wherein said recovery means includes a set of ~~correction means~~ syntax recovery units that individually employ simple functions for correcting specific types of errors; and correcting said error using

said program module by the steps of employing correction means based on the error type in accordance with the predetermined rule comprising sequentially using syntax recovery units where a syntax recovery unit receives an error message if a token does not grammatically match the syntax pointer, and the syntax recovery unit sequentially passes the error to a subsequent syntax recovery unit until a syntax recovery unit matching the error type is found, said syntax recovery unit

a) resetting the current abstract syntax tree, buffer, and context pointer;

b) recovering a pointer where in accordance with said syntax rules no error exists; and

c) notifying the parser that the correction was successfully applied., and parsing the obtained data.

13. (Original) The parsing method according to claim 12, wherein said step of selecting a program module for use includes the steps of: examining the type of said target data string; and employing said type of said target data string to select said program module based on a correlation that is defined in advance.

14. (Original) The parsing method according to claim 12, further comprising the step of: replacing, upon the receipt of an instruction from said program module to which said correction request has been issued, a rule used for said parsing with a different rule, wherein, at said step of performing said parsing for the resultant data string, said parsing is performed for said data string written in accordance with said different rule.

15. (Currently Amended) A storage medium on which input means of a computer stores a computer-readable program product that permits said computer to perform: analysis means for analyzing said data string; and recovery means, for, upon the receipt of a request from said analysis means, correcting an error detected in said data string by said analysis means, and wherein multiple types of said recovery means are prepared in accordance with the type of error that is detected by said analysis means in said data string, and each has a function for correcting a specific type of error said computer readable program causing a computer to

a) generate a grammar information object;

b) empty buffers to receive a data string;

c) cause a context pointer to indicate a root node of an abstract syntax tree;

d) obtain a set of syntax recovery units;

e) input a data string comprising a stream of tokens to the buffers;

f) extract a token from the buffer as a target token;

and when the target token is the end of the data string generating and outputting an abstract data tree; and for detecting an error in accordance with a predetermined rule by a method further comprising receiving a non-terminal token, determining from the grammar information object if the token matches a context pointer, and if so adding the token to the context pointer by the steps of:

a) adding a node to which the terminal token is appended to the context pointer;

b) shifting the point indicated by the by the context pointer to the newly added node to which the non-terminal token is appended;

c) repeating steps a) and b) for all child nodes;

d) shifting the context pointer to the parent node; and

e) obtaining a next token if the node does not contain all child nodes or if the context pointer points back to a parent node; and recovery means for, upon the receipt of a request from said analysis means, correcting in accordance with said predetermined rule said error detected in said data string by said analysis means, wherein said recovery means includes a set of ~~correction means~~ syntax recovery units that individually employ simple functions for correcting specific types of errors, and wherein said recovery means selectively employs said syntax recovery units based on the error type in accordance with said predetermined rule in order to correct a variety of errors in said data string by a method further comprising sequentially using syntax recovery units where a syntax recovery unit receives an error message if a token does not grammatically match the syntax pointer, and the syntax recovery unit sequentially passing the error to a subsequent syntax recovery unit until a syntax recovery unit matching the error type is found, said syntax recovery unit

a) resetting the current abstract syntax tree, buffer, and context pointer;

b) recovering a pointer where in accordance with said syntax rules no error exists; and

c) notifying the parser that the correction was successfully applied.

16. (Currently Amended) A program transmission apparatus comprising: storage means for storing a program product that permits a computer to perform analysis means for analyzing said data string, and recovery means, for, upon the receipt of a request from said analysis means, correcting an error detected in said data string by said analysis means; and transmission means for reading said program product from said storage means and transmitting said program product, wherein multiple types of said recovery means are prepared in accordance with the type of error that is detected by said analysis means in said data string, and each has a function for correcting a specific type of error said program product causing a computer to

a) generate a grammar information object;

b) empty buffers to receive a data string;

c) cause a context pointer to indicate a root node of an abstract syntax tree;

d) obtain a set of syntax recovery units;

e) input a data string comprising a stream of tokens to the buffers;

f) extract a token from the buffer as a target token;

and when the target token is the end of the data string generating and outputting an abstract data tree, and for detecting an error in accordance with said predetermined rule by a method further comprising receiving a non-terminal token, determining from the grammar information object if the token matches a context pointer, and if so adding the token to the context pointer by the steps of:

a) adding a node to which the terminal token is appended to the context pointer;

b) shifting the point indicated by the by the context pointer to the newly added node to which the non-terminal token is appended;

c) repeating steps a) and b) for all child nodes;

d) shifting the context pointer to the parent node; and

e) obtaining a next token if the node does not contain all child nodes or if the context pointer points back to a parent node; and recovery means for, upon the receipt of a

request from said analysis means, correcting in accordance with said predetermined rule said error detected in said data string by said analysis means, wherein said recovery means includes a set of syntax recovery units that individually employ simple functions for correcting specific types of errors, and wherein said syntax recovery means selectively employs said syntax recovery units based on the error type in accordance with said predetermined rule in order to correct a variety of errors in said data string by a method further comprising sequentially using syntax recovery units where a syntax recovery unit receives an error message if a token does not grammatically match the syntax pointer, and the syntax recovery unit sequentially passes the error to a subsequent syntax recovery unit until a syntax recovery unit matching the error type is found, said syntax recovery unit

a) resetting the current abstract syntax tree, buffer, and context pointer;

b) recovering a pointer where in accordance with said syntax rules no error exists; and

c) notifying the parser that the correction was successfully applied.